

Duplikate eliminieren

Es kann durchaus mal vorkommen, dass man ein und dieselben Daten in verschiedensten Verzeichnissen auf der Platte hat. Sowas haut die Platte voll, ohne dass wirklich mehr Info zusammen

kommt



Der erste Ansatz ist natürlich, die Duplikate zu löschen. Aber manchmal hat man die Dateien nach verschiedenen Kriterien sortiert, und es wäre unangenehm, wenn die Daten in der einen Sortierung fehlten. Deshalb habe ich mir ein Script ausgedacht, das die doppelten Dateien löscht und als Default-Aktion einen Hardlink zum Original erstellt – die Ordnung bleibt erhalten, aber es wird weniger Platz verbraucht. Per Parameter lässt sich das abstellen bzw. zum Symlink ändern.

[findup.sh](#)

```
#!/bin/bash
#####
#
# find duplicate files in a given directory
#
# written 2008-05-17
# modified for subdir handling 2008-08-10
# added parameter handling 2008-08-16
#
#####
# set -x

# default values for all operations
ACTION='HARD' # create hard links
LNPARAM='--' # parameter for the ln command
PATTERN='*' # files to look at
DEPTH=1 # directory depth
SPEAK=1 # verbosity level
E_OK=0 # exit code

# why starting the getopt argument string with a colon?
# to answer, please read 'man getopt' :-)
while getopts ":d:hlp:qsuv" Option ; do
    case $Option in
        'd' ) DEPTH=$OPTARG ;;
        'h' ) show_usage ; exit $E_OK ;;
        'l' ) ACTION='HARD' ; LNPARAM='--' ;;
        'p' ) PATTERN=$OPTARG ;;
        'q' ) SPEAK=0 ;;
        's' ) ACTION='SOFT' ; LNPARAM='-s --' ;;
        'u' ) ACTION='UN' ;;
        'v' ) SPEAK=2 ;;
    esac
done
```

```
# now we look at the other parameters (if any)
shift $(( $OPTIND - 1 ))

# if we still have a parameter, this will be the starting directory
# if the parameter is not given, it defaults to the current dir
DIR=${1-${pwd}}
# make sure we have a trailing slash
echo "$DIR" | grep '\/$' >/dev/null 2>&1 || DIR="$DIR/"

# temporary files
OUTFILE="$DIR"findsubdup.out"
test -f "$OUTFILE" && rm -f "$OUTFILE"
SORTFILE="$DIR"findsubdup.sort"
test -f "$SORTFILE" && rm -f "$OUTFILE"

function show_usage() {
    echo 'This bash script tries to eliminate duplicate files in a
    directory.'
    echo 'En passant, it removes the "executable" flag from the
    examined files.'
    echo ' '
    echo "Usage: $(basename $0) [-d n] [-h] [-l] [-p pat] [-q] [-s] [-
    u] [-v] [startdir]"
    echo '          with the following meaning of the parameters:'
    echo '          -d n      set the search depth to n directories (n is
    numeric, default: 1)'
    echo '          -h          show this help'
    echo '          -l          hardlink the duplicates to a master (default
    action)'
    echo '          -p pat      give the search pattern for duplicate files
    (default: *)'
    echo '          Please escape shell metacharacters, like -p
    "\*.jpg"'
    echo '          -q          quiet, do not echo "examined" for each
    directory'
    echo '          -s          symlink the duplicates to a master instead of
    hardlink'
    echo '          -u          unlink the duplicates (= delete them)'
    echo '          -v          verbose, echo each handled and compared file'
    echo '          startdir   where to start searching the duplicates
    (default: current directory)'
    echo ' '
}

function doonedir() {
    HELPFILE="$1"
    while read DAT ; do
        MYDAT="$DAT"
        test -x "$MYDAT" && chmod a-x -- "$MYDAT"
        MYMD5=$(md5sum -b -- "$MYDAT" | cut -d' ' -f1)
```

```

        MYSHA=$(shasum -b -- "$MYDAT" | cut -d' ' -f1)
        echo "$MYMD5 $MYSHA $MYDAT" >>"$OUTFILE"
    done < "$HELPPFILE"
}

# get all subdirs to current dir
DIRLIST=$( find "$DIR" -maxdepth $DEPTH -type d | tr ' ' '#' | tr "\n"
" ")

for WDIR in $DIRLIST ; do
    # get all files in the current dir
    # Problem: may be too many files, so better store them in a help
    file
    WODIR=$(echo "$WDIR" | sed 's/#/\ /g')
    HELPPFILE="${WODIR}/findsubdup.help"
    test -f "$HELPPFILE" && rm -f -- "$HELPPFILE"

    # just search in the current directory, don't use $DEPTH here!
    find "$WODIR" -maxdepth 1 -type f -name "$PATTERN" -fprintf
"$HELPPFILE" "%p\n"
    # now loop through the resulting file and compute MD5sum and
    SHA1sum
    # again, the result is put into a file
    doonedir "$HELPPFILE"

    rm -f -- "$HELPPFILE"
    test $SPEAK -ge 1 && echo "[$WODIR]: examined"
done

# sort the resulting file on MD5sum
sort <"$OUTFILE" >"$SORTFILE"
rm -f -- "$OUTFILE"

# walk through the sorted file and compare MD5sums/SHA1sums
# when they are equal, echo this and delete one of them
OLD_FL="none"
OLD_MD="init"
OLD_SH="init"
DELCOUNT=0
while read MD SH FL ; do
    if [ "x$MD" = "x$OLD_MD" -a "x$SH" = "x$OLD_SH" ]; then
        test $SPEAK -eq 2 && echo "same MD5sum/SHA1sum for [$OLD_FL]
and [$FL], deleting [$FL]"
        rm -f -- "$FL" && let "DELCOUNT += 1"
        test "$ACTION" != 'UN' && ln $LNPARAM "$OLD_FL" "$FL"
    else
        OLD_MD="$MD"
        OLD_SH="$SH"
        OLD_FL="$FL"
    fi
done < "$SORTFILE"

```

```
rm -f -- "$SORTFILE"  
echo "Worked on $DELCOUNT duplicates, exiting now"  
exit $E_OK
```

Ein dickes Dankeschön an Mendel Cooper dafür, dass er den großartigen [Advanced Bash-Scripting Guide](#) geschrieben hat!

From:

<http://wernerflamme.net/> - **Werners Wiki**

Permanent link:

<http://wernerflamme.net/doku.php?id=comp:deldup>

Last update: **2010-11-03 1255**

